Penn Engineering
ONLINE LEARNING

CIT 591 Introduction to Software Development

Module 1: Your First Java Program

Module Learning Objectives

- Set up a development environment
- Write a basic program that reads from and writes to the console
- Create your own class
- Discriminate between useful and non-useful comments
- Write code that another intro programmer can read and understand

Module Glossary

- **camelCasing:** A way of naming variables/methods such that each word after the first begins with a capital letter and there is no punctuation in between the different words. For example studentStudyHall is camel cased and proper Java convention.
- **Classes and Objects:** An object is an instance of a class. Think of a class as a template (a mold, a blueprint) and think of objects as the results of using that template. Human is a class. You and I are objects of type Human.
- **Concatenation:** The operation of joining character strings. For example, the concatenation of "comp" and "ute" is "compute".
- **Datatype:** A data type or simply type is an attribute of data which tells the compiler or interpreter how the programmer intends to use the data.
- **IDE:** Integrated Development Environment - an application that provides a suite of software development tools including code editing and debugging. In this course, we use Eclipse.
- **JDK:** Java Development Kit, the software package you need to install on your computer to be able to program in Java.
- **Method:** A collection of Java code statements that perform some “work”. From an object-oriented standpoint, these correspond to the things that your objects do.
- **Parameters:** Parameters are used to pass information to a method.
- **Return:** A *return statement* causes execution to exit the current method and provide the value of a variable to the point in the code where this method was called.

- **Scanner:** An inbuilt Java class used for scanning simple text. Can parse primitive types and strings using regular expressions (more on this in a few weeks).
- **Scope:** The set of lines of code where a variable can be accessed or used. A variable is scoped to the block of code that it is defined in.
- **String:** String is a datatype used to represent a sequence of characters.
- **Variable:** A variable is a storage location in a program. Each variable has a name and holds a value.
- **Void:** Void is the keyword used to notify the compiler that this method is not returning anything.

Module Resources

- **Textbook Readings:**
  - Chapter 1:Introduction
  - 2.1: Objects and Classes
  - 2.2: Variables
  - 2.3: Calling Methods
  - 3.1: Instance Variables and Encapsulation
  - 3.2: Specifying the Public Interface of a Class
  - 3.3: Providing the Class Implementation
  - 4.3.1: Input and Output
  - 4.5: Strings (4.5.4 and 4.5.5 are not required, but are useful to explain upcoming advanced topics)
    *Note: While you will see the keyword **private** in this chapter, we cover that in a later module. For now, don't worry about public, private etc.*
  - 4.1: Numbers
  - 4.2: Arithmetic
- **Websites:**
  - Java Development Kit (JDK) Website
  - Download Eclipse (IDE for Java)

Key Concepts & Examples

Hello World program: The traditional first program is to print a message to the console.

- Example used in the video: "Hello, world!"

Scanner: Reading simple text and saving it to a variable. The Scanner is a great option for parsing input provided by a user. We will explore reading in larger sets of data when we cover file handling in a few weeks.

- Example used in the video: asking a user for their name so that we can customize our “Hello World” message.

Instance Variables and Data Types: A variable is a storage location in a program. Each variable has a name and holds a value. A **data type** or simply **type** is an attribute of data which tells the compiler or interpreter how the programmer intends to use the data. Different languages have different datatypes. The ones you need to know in this course are int, double, boolean, char, and String. See subsequent modules for one big difference between String and those other datatypes.

- Examples used in the video: A car has various properties and states that can be converted to instance variables, such as color and current speed. We also looked at doing basic mathematical operations using variables.

Methods: A collection of Java code statements that perform some “work”. From an object-oriented standpoint, these correspond to the things that your objects do. In other words, “methods” of interacting with your object. For example, a BankAccount class is likely to have a deposit method. A Library is likely to have a searchCatalogue method. For more details (definitely more than you need at this stage) see the official documentation.

Parameters are used to pass information to a method - as described in the official Java documentation, “*Parameters* refers to the list of variables in a method declaration. *Arguments* are the actual values that are passed in when the method is invoked.” When you invoke the method, make sure your arguments match the original type and order with which you declared the parameters.

- Examples used in the video: We looked at the “verbs” associated with the car (drive), writing code for a `drive` method including the method’s parameters such as speed and direction.

Comments & Style: Following style conventions and using good comments makes it easier to read and debug your code as you’re writing it, as well as reading and

maintaining/upgrading your code in the future. Get in the habit of using these conventions in your code!

Assignment 1: Strings & Classes: Now that you have learned the basics of classes and methods as well as how to use variables to store data of various types, you're ready to put these skills into action! The activities in Module 1 Recitation will help you practice for this assignment, so it is highly recommended that you attend the live session or review the recordings and be sure to attempt the recitation activities.